



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/657,579	09/08/2003	B. Martin Hill	IDF 2553 (4000-09001)	9871
28003	7590	09/04/2008		
SPRINT 6391 SPRINT PARKWAY KSOPHT0101-Z2100 OVERLAND PARK, KS 66251-2100			EXAMINER ANYA, CHARLES E	
			ART UNIT 2194	PAPER NUMBER
			MAIL DATE 09/04/2008	DELIVERY MODE PAPER

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.



### DETAILED ACTION

1. Claims 1-37 are pending this application.

#### ***Claim Rejections - 35 USC § 103***

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

2. **Claims 1-10, 12, 13, 15-23 and 31-37 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pub. No. 7,159,579 B2 issued to Sharma et al. in view of U.S. Pub. No. 20020032783 A1 issued to Tuatini.**

3. As to claim 1, Sharma teaches a method of accessing an Enterprise Java Bean (EJB) by an application within a computing environment, comprising: a) making a call by the application (Client Side Runtime System 134, "...client side proxy..." Col. 7 Ln. 56 – 67, "...javax.xml.rpc.Serviceinterface...dynamically proxy..." Col. 8 Ln.12 – 53); b) invoking a function within the client library to construct an HTTP request corresponding to the input parameters of the call from the application (Client Side Runtime System 525 "Steps 650/670" Col. 23 Ln. 16 – 33, Ln. 52 – 58); c) passing the HTTP request from the client library to an EjbServlet/d) invoking a method on an EJB by the EjbServlet based upon the HTTP request ("...servlet container...HTTP requests...call a method..." Col. 6

Art Unit: 2194

Ln. 1 – 26, "...transported..." Col. 23 Ln. 58 – 61, Step 740 Col. 24 Ln. 21 – 31); e) returning information from the invoked method from the EJB to the EjbServlet ("... component's generation of a response..." Col. 6 Ln. 1 – 12, "...generate response..." Col. 24 Ln. 30 – 40); decoding the returned information into an HTTP response string by the EjbServlet ("...produces a response..." Col. 6 Ln. 16 – 17, "...deserializers..." Col. 22 Ln. 5 – 8); and g) transmitting the HTTP response from the EjbServlet to the client library ("...serve the content..." Col. 6 Ln. 21 – 26, Container 560 "Steps 750/760" Col. 24 Ln. 33 – 40).

Sharma is silent with reference to a non-Java application and parsing and converting the HTTP response by the client library into return information compatible with the non-Java application and then passing the return information from the client library to the non-Java application.

Tuatini teaches a non-Java application ("...shared services and clients can also include...non-Java components page 14 paragraph 0120, "...legacy external application..." page 15 paragraph 0130) and parsing and converting the HTTP response by the client library into return information compatible with the non-Java application and then passing the return information from the client library to the non-Java application ("...formatting the responses..." page 2 paragraph 0060, View Layer 307 page 3 paragraph 0063, "...appropriate format..." page 14 paragraph 0119, "...DTD for zero or more response messages..." page 15 paragraph 0127, "...translation..." page 16 paragraph 0138).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the system of Sharma with the teaching of Tuatini because the teaching of Tuatini would improve the system of Sharma by eliminating the costs associated with the redesigning of the messaging system by allowing for some form of emulation or backward compatibility to custom-designed and legacy applications (Tuatini page 14 paragraph 0120).

4. As to claim 2, Sharma teaches the method of claim 1 wherein the invoked method is a chained method (“...remote method call...remote call...” “Step 740” Col. 23 Ln. 53 – 67, Col. 24 Ln. 14 – 32).

5. As to claim 3, Sharma teaches the method of claim 1 wherein the invoking a method on the EJB by the EjbServlet further comprises passing one or more input parameters by the EjbServlet to the EJB (“...provide context data (i.e., session state and information about the request...” Col. 6 Ln. 8 – 13).

6. As to claim 4, Sharma teaches the method of claim 3 wherein the input parameters comprise one or more input objects constructed by the EjbServlet based on the HTTP request (“...provide context data (i.e., session state and information about the request...” Col. 6 Ln. 8 – 13).

Art Unit: 2194

7. As to claim 5, Sharma teaches the method of claim 1 wherein the returning information from the invoked method from the EJB to the EjbServlet further comprises the EJB constructing a return object based upon the information from the invoked method and passing the return object to the EjbServlet (“...produces a response...” Col. 6 Ln. 11 – 17).

8. As to claim 6, Sharma teaches the method of claim 5 wherein the return object is a chained object (“...the container may return the content...serve the content...” Col. 6 Ln. 11 – 26).

9. As to claim 7, Sharma teaches the method of claim 5 wherein the decoding the returned information into an HTTP response string by the EjbServlet further comprises decoding the return object into an HTTP response string by the EjbServlet (“...produces a response...” Col. 6 Ln. 16 – 17, “...deserializers...” Col. 22 Ln. 5 – 8).

10. As to claim 8, Sharma teaches the method of claim 1 wherein the HTTP request and the HTTP response each comprise a series of bytes representing HTTP-specific control information or text strings (“...HTTP request...” Col. 23 Ln. 52 – 67, “...HTTP response...” Col. 24 Ln. 32 – 40).

11. As to claim 9, Sharma teaches the method of claim 8 wherein the HTTP request and the HTTP response are passed between the client library and the EjbServlet via an

Art Unit: 2194

HTTP protocol (Col. 6 Ln. 1 – 26, “...HTTP request...HTTP transport...” Col. 23 Ln. 52 – 67, “...HTTP response...” Col. 24 Ln. 32 – 40).

12. As to claim 10, Sharma teaches the method of claim 9 wherein the HTTP protocol enables the client library and EjbServlet to communicate across a distributed computing environment (Col. 6 Ln. 1 – 26, “...HTTP request...HTTP transport...” Col. 23 Ln. 52 – 67, “...HTTP response...” Col. 24 Ln. 32 – 40).

13. As to claim 12, Tuatini teaches the method of claim 1 further comprising the non-Java application allocating buffers to hold calling input parameters sent to the client library and return information received from the client library (“...asynchronous...” page 15 paragraph 0131).

14. As to claim 13, Sharma teaches the method of claim 1 wherein the client library is a linkable library (“...JAX-RPC library...” Col. 7 Ln. 31 – 55).

15. As to claim 15, Sharma teaches the method of claim 1 further comprising the client library converting any numeric input parameter in the calling input parameters into a corresponding text representation in the HTTP request (“...translate requests...” Col. 6 Ln. 1 – 26, “...mapping...” Col. 23 Ln. 16 – 33).

Art Unit: 2194

16. As to claim 16, Sharma teaches the method of claim 12 further comprising the client library extracting the return information from the HTTP response sent by the EjbServlet and placing the return information into the buffers provided by the calling application (“...translate requests...” Col. 6 Ln. 1 – 26, “...mapping...” Col. 23 Ln. 16 – 33).

17. As to claim 17, Sharma teaches the method of claim 16 further comprising the client library converting any text-represented numeric value extracted from the HTTP response into a corresponding numeric form thereof (“...translate requests...” Col. 6 Ln. 1 – 26, “...mapping...” Col. 23 Ln. 16 – 33).

18. As to claim 18, Sharma teaches the method of claim 1 wherein the EjbServlet and the method invoked on the EJB is identified by a calling input parameter embedded in the HTTP request (“...call a method...” Col. 6 Ln. 17 – 24).

19. As to claim 19, Tuatini teaches the method of claim 1 wherein the method is invoked via a remote method invocation (RMI) protocol (“...Remote Method Invocation (RMI)...” page 16 paragraph 0134).

20. As to claim 20, Sharma teaches the method of claim 19 wherein the RMI enables the EjbServlet and the EJB to communicate across a distributed computing environment (“...RMI...” page 6 paragraph 0091, page 10 paragraph 0193).



21. As to claim 21, Tuatini teaches the method of claim 12 wherein the return information is placed into two buffers (“...asynchronous...” page 15 paragraph 0131).

22. As to claim 22, Tuatini teaches the method of claim 21 wherein the buffers comprise a data buffer and a format buffer. (“...asynchronous...” page 15 paragraph 0131).

23. As to claim 23, Sharma teaches the method of claim 22 further comprising passing additional decoded return information wherein the information from the invoked method exceeds the data buffer capacity, the format buffer capacity, or both (“...deserializers...” Col. 23 Ln. 8 – 9).

24. As to claim 31, Tuatini teaches the method of claim 1 further comprising invoking a logging function within the client library (“logging...” page 4 paragraph 0067).

25. As to claims 32,36 and 37, see the rejection of claim 1 above.

26. As to claims 33-35, see the rejection of claims 3, 5 and 8 respectively.

Art Unit: 2194

**27. Claims 11 and 14 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pub. No. 7,159,579 B2 issued to Sharma et al. in view of U.S. Pub. No. 20020032783 A1 issued to Tuatini as applied to claim 23 above, and further in view of U.S. Pub. No. 2005/0223392 A1 issued to Cox et al.**

28. As to claim 11, Tuatini and Sharma are silent with respect to the method of claim 1 wherein the non-Java application is based on a programming environment capable of calling external library functions via the C calling convention (“...Visual C++...C# language...” page 4 paragraph 004, page 8 paragraph 0174).

Cox teaches the method of claim 1 wherein the non-Java application is based on a programming environment capable of calling external library functions via the C calling convention (“...Visual C++...C# language...” page 4 paragraph 004, page 8 paragraph 0174).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the system of Tuatini and Sharma with the teaching of Cox because the teaching of Cox would improve the system of Tuatini and Sharma by providing a general-purpose programming language with high-level and low-level capabilities and functionalities, including being statically typed, free-formed, multi-paradigm, usually compiled language and supporting procedural programming, data abstraction, object-oriented and generic programming.

Art Unit: 2194

29. As to claim 14, Cox teaches the method of claim 1 wherein the calls between the client library and the non-Java application are based upon the C language calling convention (“...Visual C++...C# language...” page 4 paragraph 004, page 8 paragraph 0174).

**30. Claims 24-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pub. No. 7,159,579 B2 issued to Sharma et al. in view of U.S. Pub. No. 20020032783 A1 issued to Tautini as applied to claim 23 above, and further in view of U.S. Pub. No. 2006/0036448 A1 issued to Haynie et al.**

31. As to claim 24, Tautini and Sharma are silent with reference to the method of claim 23 wherein the EjbServlet stores the remaining decoded EJB method call results in memory.

Haynie teaches the method of claim 23 wherein the EjbServlet stores the remaining decoded EJB method call results in memory (“...caching...” page 11 paragraph 0149).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the system of Tautini and Sharma with the teaching of Haynie because the teaching of Haynie would improve the system of Tautini and Sharma by providing a temporary storage area where frequently accessed data can be stored for rapid access.

Art Unit: 2194

32. As to claim 25, Haynie teaches the method of claim 23 wherein the client library passes a return code to the non-Java application indicating that information from the invoked method remains in the EjbServlet (“...METHOD CALL RESPONSE...” page 11 paragraph 0154).

33. As to claim 26, Haynie teaches the method of claim 23 wherein additional return data and format strings are passed until all of the decoded information from the invoked method is received by the application (“...METHOD CALL RESPONSE...” page 11 paragraph 0154).

34. As to claim 27, Haynie teaches the method of claim 23 wherein the EjbServlet passes a key to the client library identifying any information from the invoked method remaining in the EjbServlet (“...object ID parameter...” page 11 paragraph 0149).

35. As to claim 28, Haynie teaches the method of claim 27 wherein the client library places the key in a session ID parameter (“...object ID parameter...” page 11 paragraph 0149).

36. As to claim 29, Haynie teaches the method of claim 28 wherein the client library provides the key to the non-Java application (“...object ID parameter...” page 11 paragraph 0149).

Art Unit: 2194

37. As to claim 30, Haynie teaches the method of claim 29 wherein non-Java application accesses the information from the invoked method remaining in the EjbServlet using the key (“...object ID parameter...” page 11 paragraph 0149).

### ***Response to Arguments***

Applicant's arguments filed 5/27/08 have been fully considered but they are not persuasive.

Applicant argues in substance that (1) the Sharma and Tuatini prior arts do teach or suggest a client library, more so because the office action does not indicate which element(s)/component(s) are being read as client library, (2) the Tuatini prior art does not teach or suggest parsing and converting Http response by the client library, and (3) the Sharma and Tuatini prior arts do not teach or suggest a non-Java application.

The Examiner respectfully traverses Applicant's arguments:

As to point (1), firstly, the fact that more than one element/component maps to the claimed client library does not negate the fact that the functionality of the claimed client library are provided by the Sharma prior art. The client runtime system (Client Side Runtime System 134/Client-Side Runtime System 525) and client side application programming interface (Client Side API(s) 135/Stub 515/Core APIs 520) of the Sharma prior art provides the functionality of the claimed client library.

Sharma prior art discloses distributed computing systems and web services, and more particularly, processes for providing a servlet container based web service endpoint in a remote procedure call based on the distributed computing system. The

Art Unit: 2194

distributed computing system includes a client (Client 130) that uses “JAX-RPC” based mechanism to generate and communicate calls to server (Server 110/servlet container). The client (Client Application 138/Service Client Application 510) generates a call request to invoke one or more methods on the defined service endpoint (Server 110) based on the information contained in the imported WSDL document. The **call request** (making a call/invoking a function) is **received** by a runtime system (Client Side Runtime System 134/Client-Side Runtime System 525) that remotely **instantiates** and initializes a service endpoint object associated with the defined service endpoint (Server 110). Once initialized, the runtime system invoke one or more methods associated with the call request on the service endpoint object (passing the HTTP request/invoking a method) and receive result data based on the invocation. The result data may be configured in a response message (returning information/transmitting the HTTP response) that is provided to the client using the same protocol and transport binding the call request. As for Client side API 135/Stub 515/Core APIs 520, they are one or more programming interfaces that enable client side JAX-RPC runtime system 134 to communicate with other processes operating in client 130. The client API 135/Stub 515/Core APIs 520 may include a javax.xml.rpc.Stub interface, a javax.xml.rpc.Call interface, a javax.xml.rpc.Service interface, a javax.xml.rpc.ServiceFactory class, and a java.xml.rpc.JAXRPCException class. The javax.xml.rpc.Stub interface is implemented by stub classes that are generated by a mapping tool operating within client 130.

It is therefore logical to conclude that the client side runtime system (Client Side Runtime System 134/Client-Side Runtime System 525) in conjunction with client side

Art Unit: 2194

application programming interface (Client Side API(s) 135/Stub 515/Core APIs 520) provides the functionality of the claimed client library.

As to point (2), the Tuatini prior art discloses an application architecture for developing applications for a computer system. The application architecture provides modules within an application that can inter-communicate and such that multiple applications can inter-communicate. The application architecture includes an application framework and applications. The application framework receives requests for services or functionality from client computers, identifies the action handlers that can service the requests, invokes the identified action handlers to service the requests to generate responses, **identifies view handlers for formatting the responses, and invokes identified view handlers to format and send the responses to the client computers**. The action handlers may also indicate a presentation view that specifies the way in which the responses are to be presented to the client systems.

The application framework as the claimed client library via the view layer (View Layer 37) translates or formats responses and sends the response to the client computers.

As to point (3), the client computers (for requesting services or making calls) of the Tautini prior art includes both commercial software and custom-designed software, and can include both new and **legacy applications**. The client computers also include applications supporting various languages and technologies (e.g., non-Java components).

Legacy applications are old application program that continues to be used because the user does not want to replace or redesign it. The legacy applications include mainframe applications written many years ago in languages such as COBOL, PL/1, Assembler, FORTRAN, and REXX and therefore non Java applications.

### ***Conclusion***

**THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Charles E. Anya whose telephone number is 571-272-3757. The examiner can normally be reached on 8:30-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on 571-272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.



Art Unit: 2194

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

cea.

/Li B. Zhen/  
Primary Examiner, Art Unit 2194